Principles of Computer Game Design and Implementation

Lecture 12

We already knew

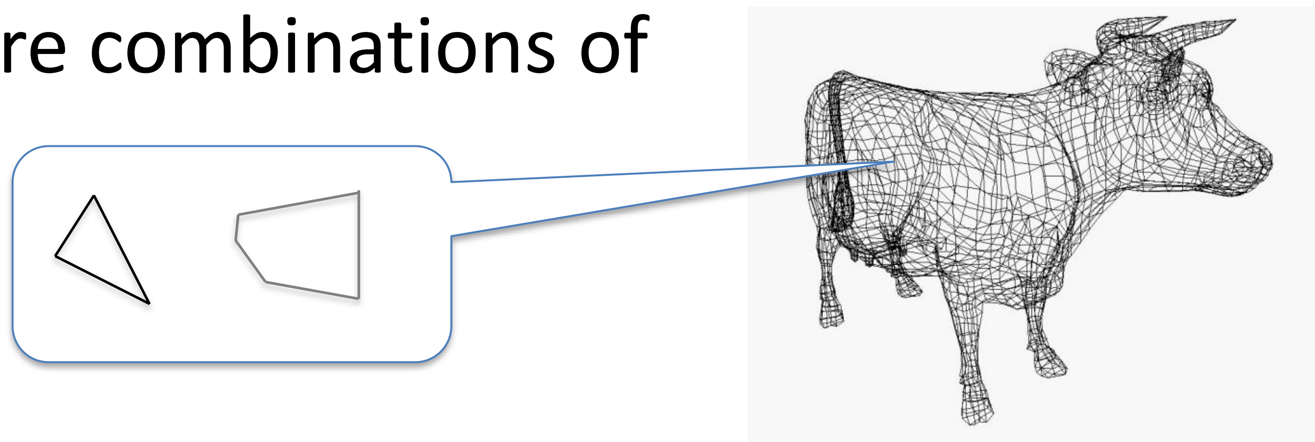
- Vector operations
- Collision detection – overlap test and intersection test

Outline for Today

- Collision detection – detailed view
- Collision detection -- mid level view

Detailed View

- 3D shapes are combinations of *polygons*

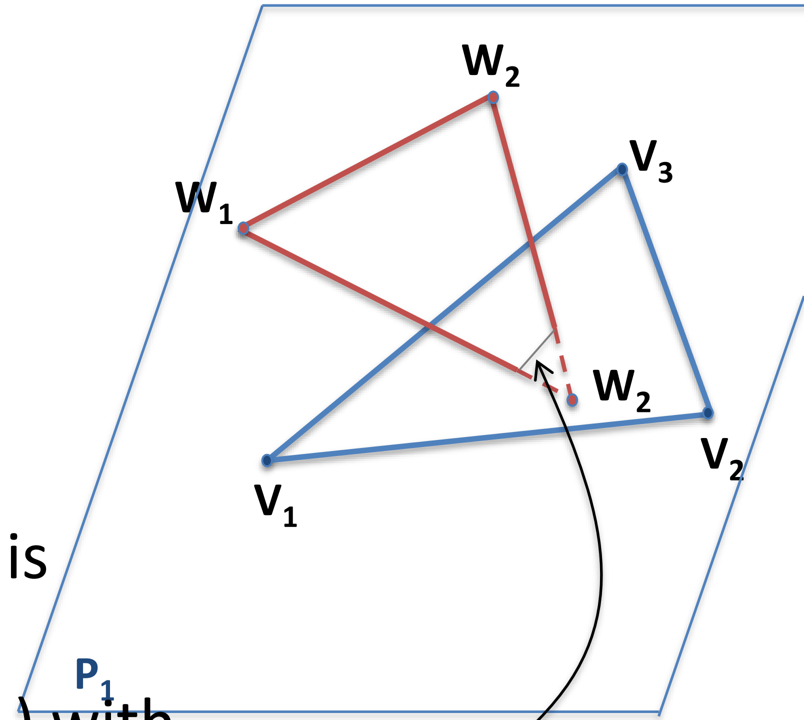


- One needs to know if
  - one polygon overlaps with another
    - Overlap testing
  - a polygon overlaps with a shape
    - Intersection testing

Overlap of Triangles

The penetration method

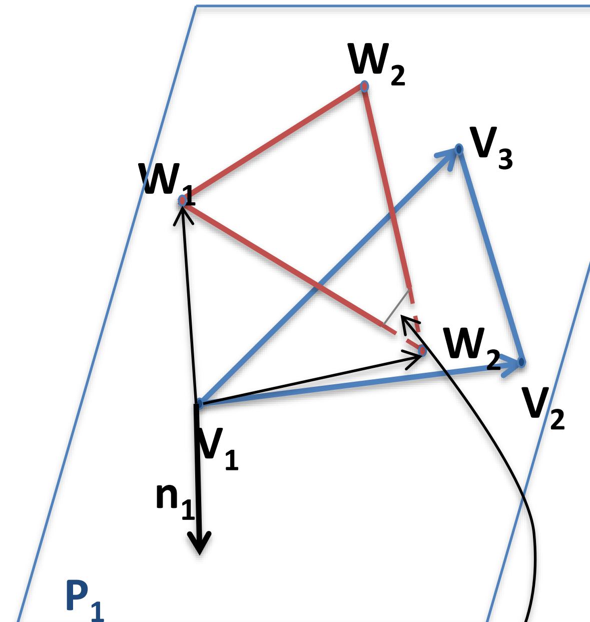
- Consider a plane $\mathbf{P_1}$ where (V_1,V_2,V_3) lays
- Triangles intersect if
  - One of $\mathbf{W}_i$ is above $\mathbf{P_1}$ and one is below
  - The intersection of $(\mathbf{W}_1,\mathbf{W}_2,\mathbf{W}_3)$ with $\mathbf{P_1}$ (line segment) lays within (V_1,V_2,V_3)



Example: Triangle & Plain

- Compute the *normal vector*

$$\mathbf{n}_1 = (\mathbf{V}_3 - \mathbf{V}_1) \times (\mathbf{V}_2 - \mathbf{V}_1)$$



- Notice that

$$\mathbf{n}_1 \cdot (\mathbf{W}_1 - \mathbf{V}_1) < 0$$

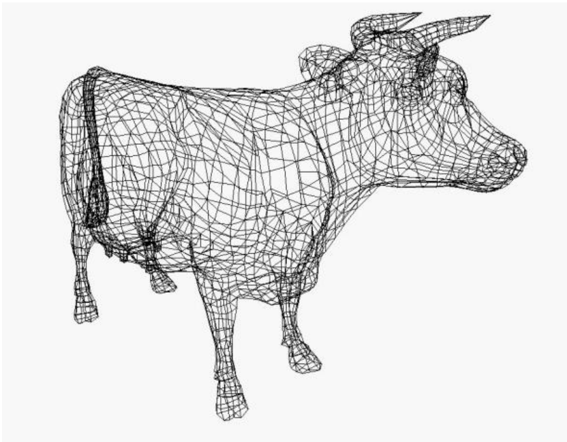
- but

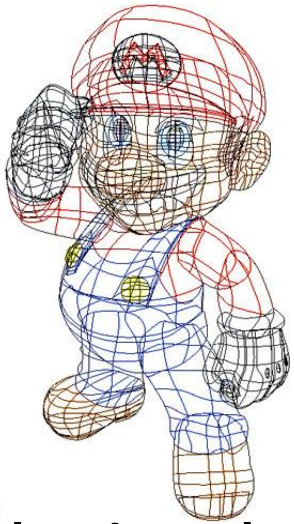
$$\mathbf{n}_1 \cdot (\mathbf{W}_2 - \mathbf{V}_1) > 0$$

Checking for requires equations for plain and line intersection

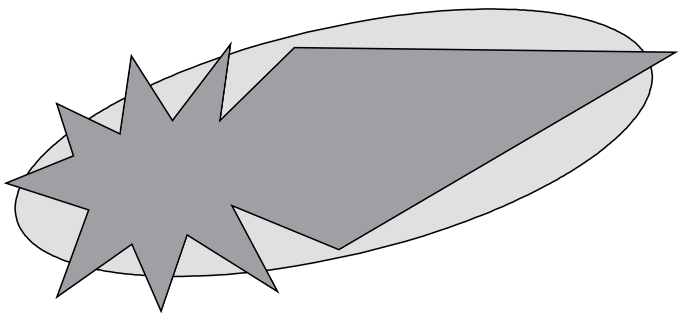
Bounding volumes

Collision detection for triangles is insanely complex for real objects

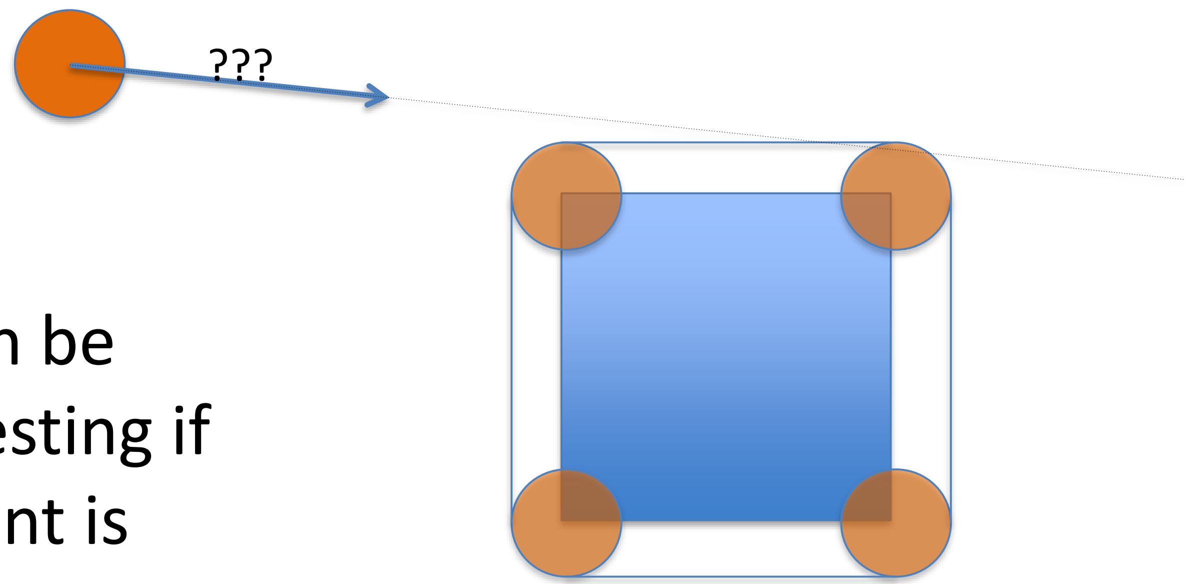




- Approximate complex objects with simpler geometry



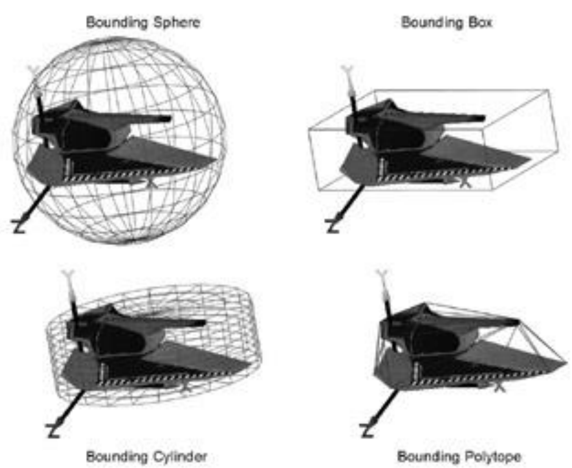
Uses: Minkowski Sum



Overlap can be found by testing if a single point is within the new volume

Uses: Bounding Volumes

- Bounding volume is a simple geometric shape
  - Completely encapsulates object
  - If no collision with bounding volume, no more testing is required
- Common bounding volumes
  - Sphere
  - Box

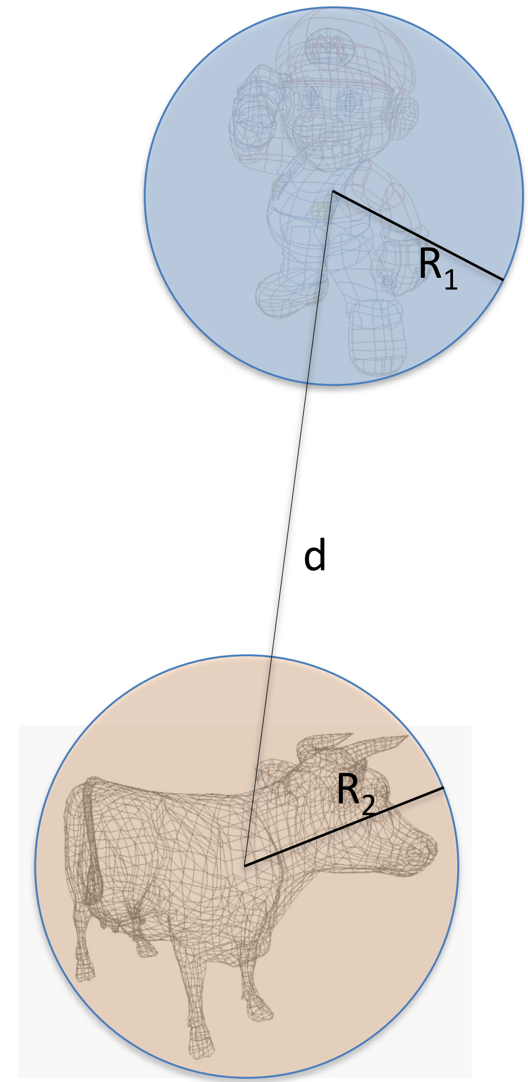


Bounding Sphere

- Simple shape approximation
  - May be difficult to get it tight
  - Two sphere collision:
    - Let $\mathbf{V}_1$ and $\mathbf{V}_2$ be *position vectors*
    - If $d < R_1 + R_2$ they overlap
      - where $d = |\mathbf{V}_1 - \mathbf{V}_2|$
    - Or, better, if

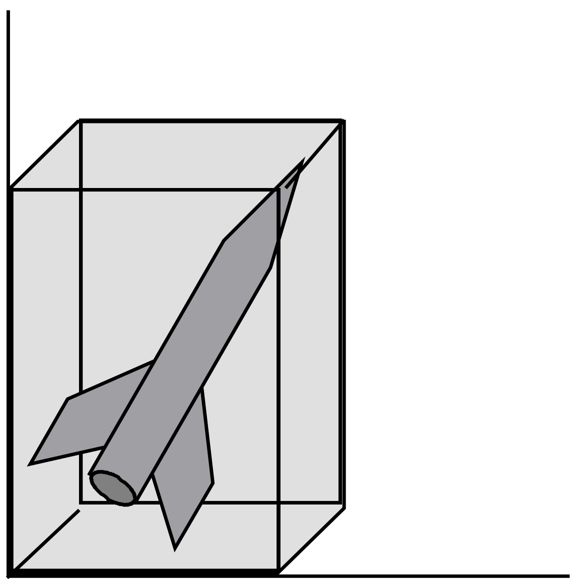
$$d^2 < (R_1 + R_2)^2$$

Why is it better?

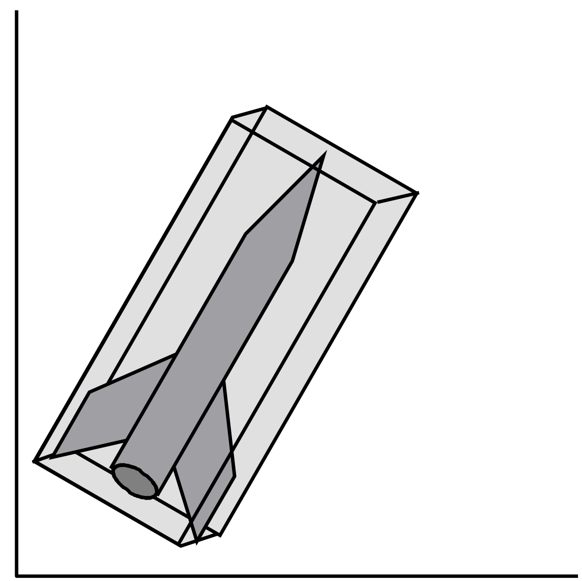


Bounding Boxes

- Place a box around an object
- Test collisions between the boxes



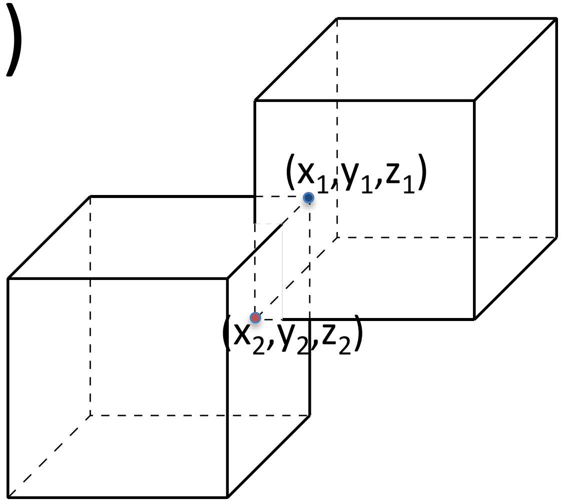
Axis-Aligned Bounding Box
(AABB)



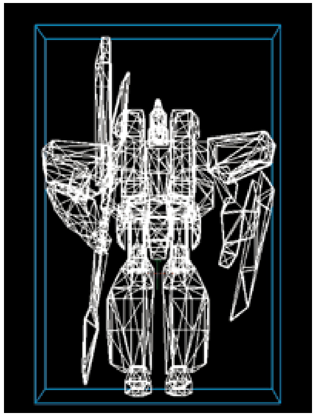
Oriented Bounding Box
(OBB)

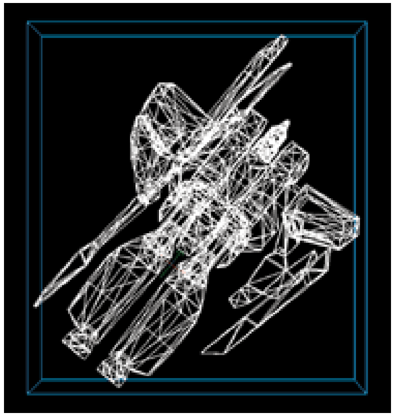
Axis-Aligned Bounding Box

- Take the maximal and minimal values of the coordinates (corner of the box)
- Collision detection is very Fast
    - Compare the corner coordinates



- May not be accurate

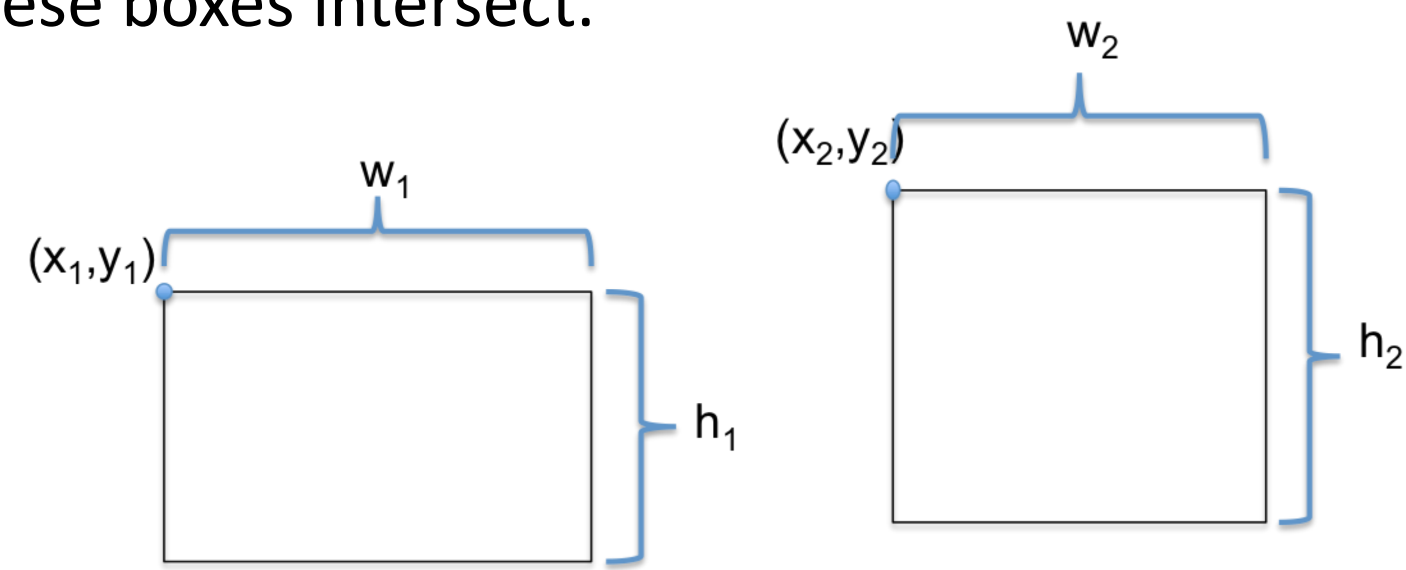




Box stretches as the object rotates

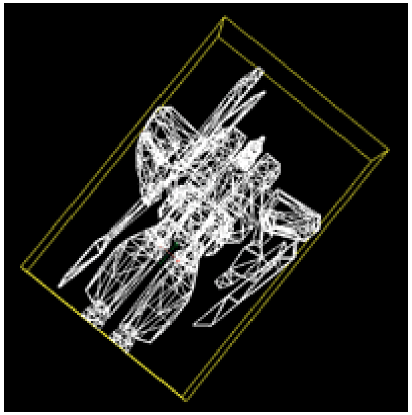
Quiz

- Sketch a method which, given the coordinates of upper left corners of two 2-dimensional axis-aligned boxes (x1,y1) and (x2,y2) and their width w1, w2 and height h1, h2, respectively, determines whether these boxes intersect.

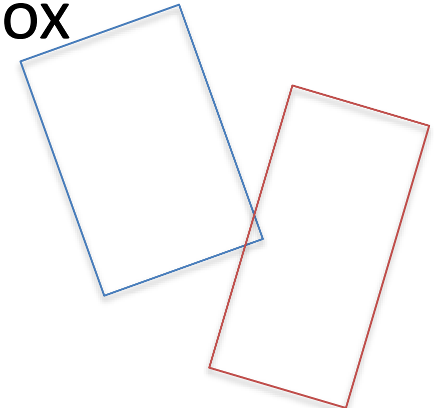


Oriented Bounding Box

- Based on object primary dimensions
- More accurate
- Box rotates with the object

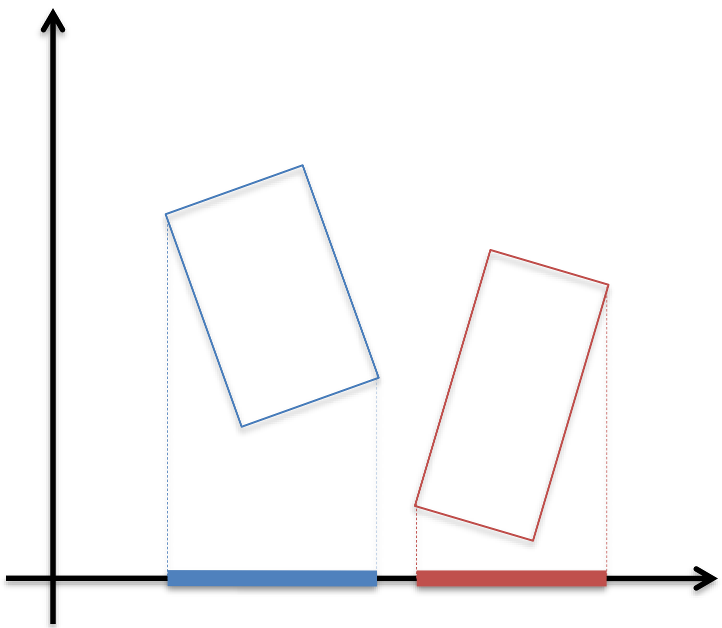


- Collision detection is harder
  - One needs to know if a point "goes across" a side of the box

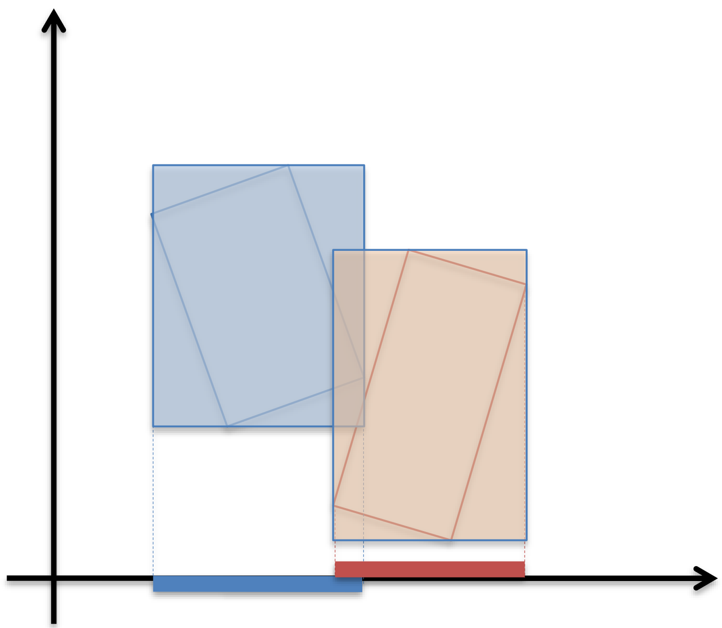


Idea: Separate Boxes

- Approximate using projections



Definitely do not overlap



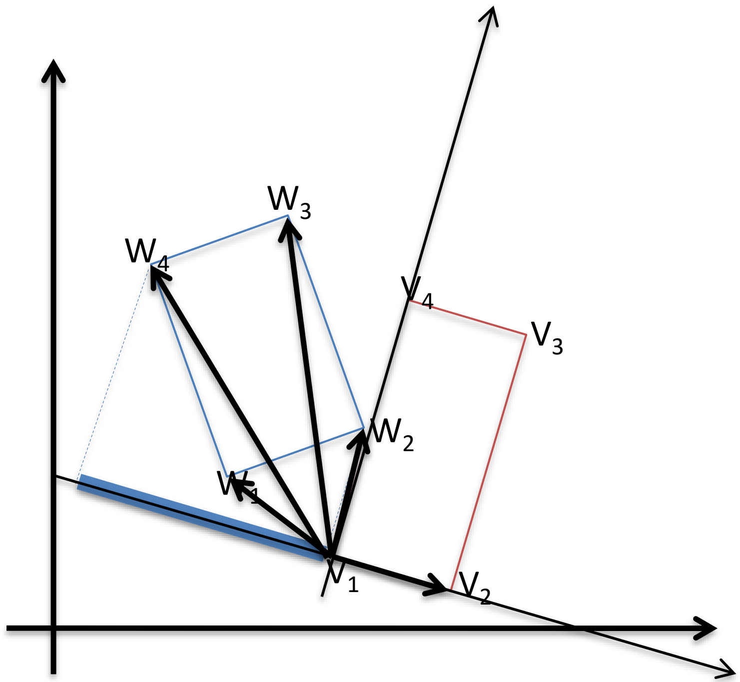
Possibly overlap
Further check needed

Nothing but putting OBBs inside AABBs

Definite Answer with Projections

- Use local coordinates given by the box edges

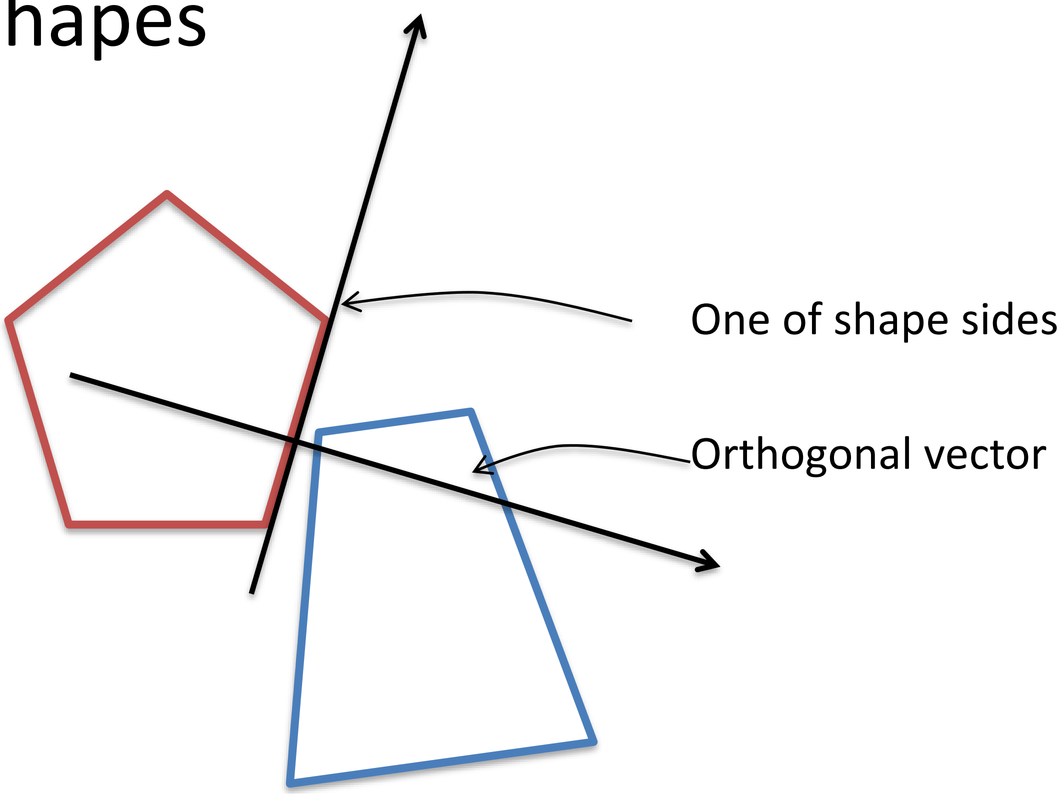
$$(\mathbf{V}_2 - \mathbf{V}_1) \cdot (\mathbf{W}_1 - \mathbf{V}_1) < 0$$
$$(\mathbf{V}_2 - \mathbf{V}_1) \cdot (\mathbf{W}_2 - \mathbf{V}_1) < 0$$
$$(\mathbf{V}_2 - \mathbf{V}_1) \cdot (\mathbf{W}_3 - \mathbf{V}_1) < 0$$
$$(\mathbf{V}_2 - \mathbf{V}_1) \cdot (\mathbf{W}_4 - \mathbf{V}_1) < 0$$

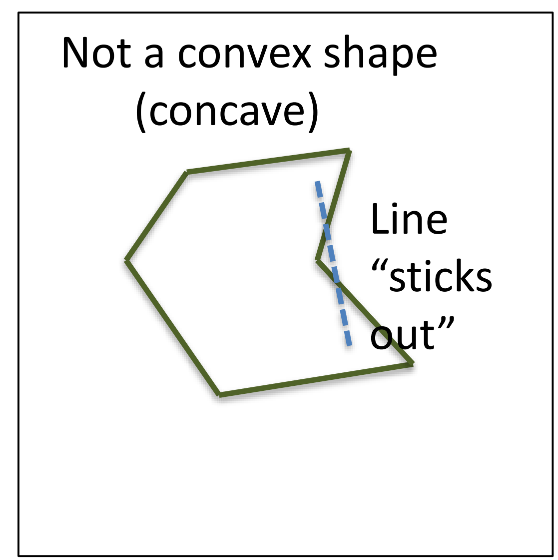


All **W**'s are to the left of ($\mathbf{V}_1 - \mathbf{V}_4$)

Separating Shapes

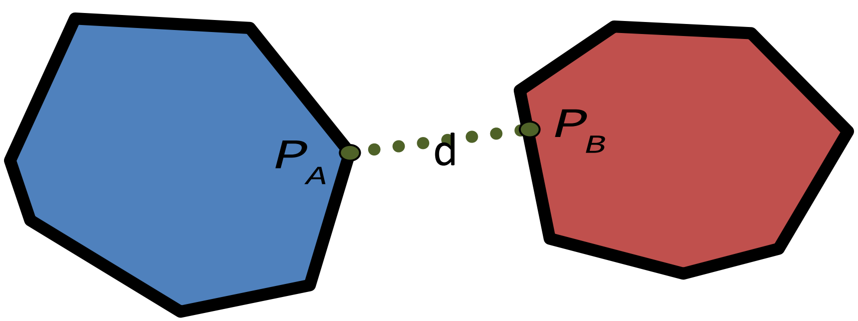
- Same principles can be applied to check for collision of arbitrary <u>*convex*</u> shapes





Distance Test

- Gilbert-Johnson-Keerthi (GJK) Algorithm
  - Determines *distance* between two convex shapes
  - Can be used to locate closest points
  - Uses Minkowski sum
  - Requires some maths background to understand
  - Implementations available



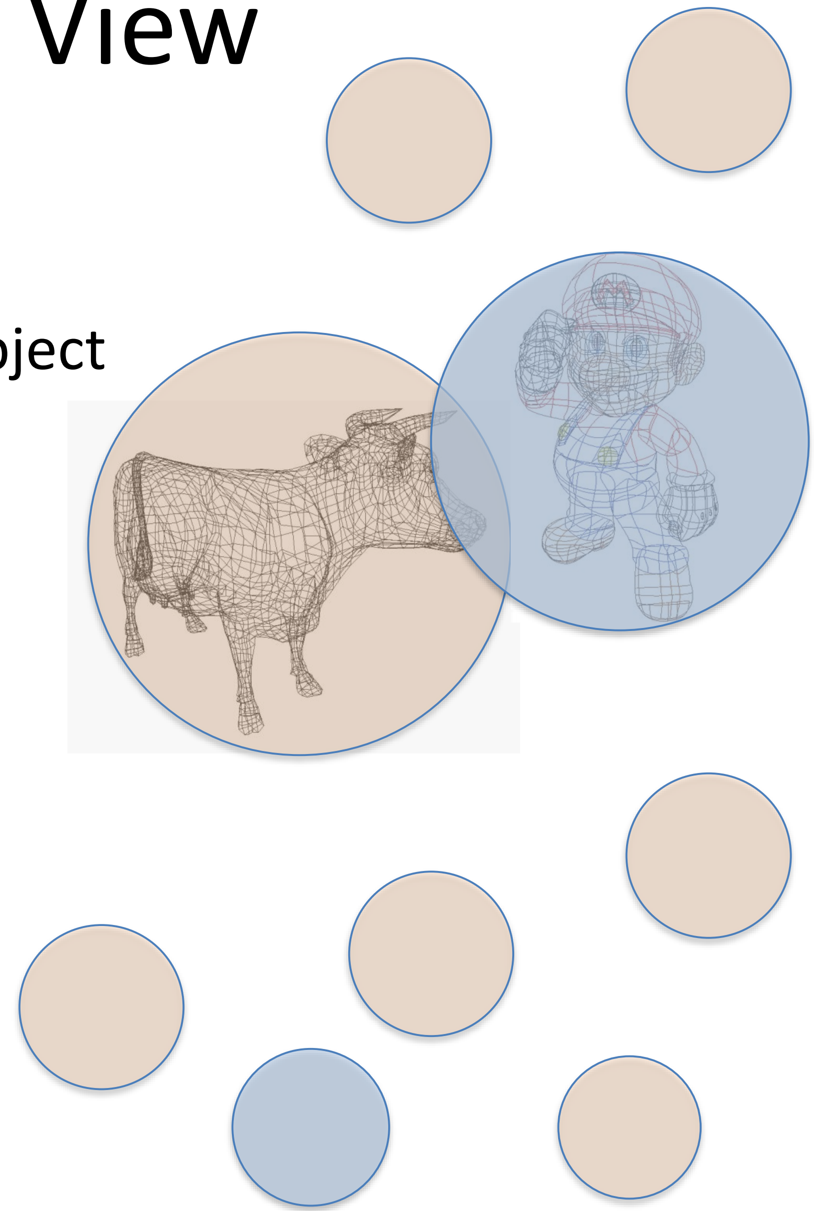
Mid-Level View

Speedup recipe:

- Place a simple shape around an object
- Test for collisions between the bounding shapes

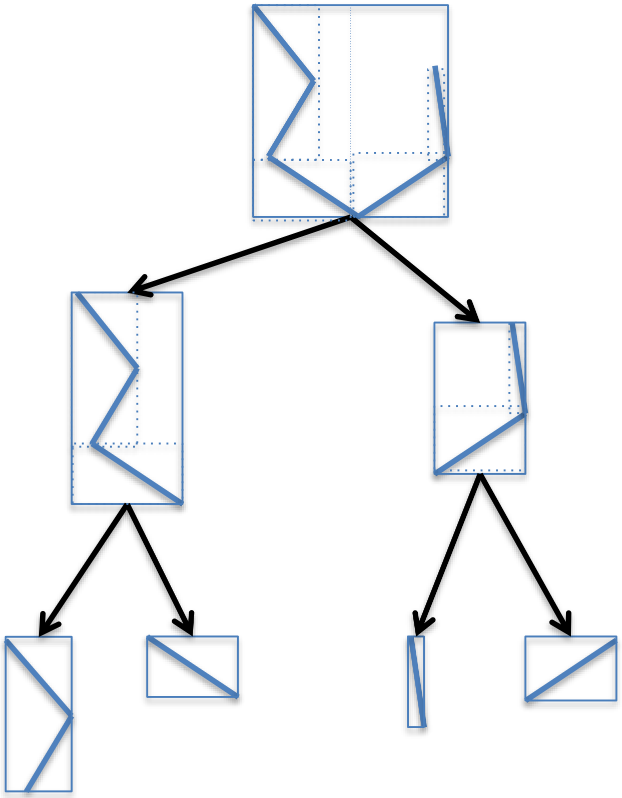
- Two problems:
  - Too crude an approximation
  - Too many entities

- Divide and conquer!



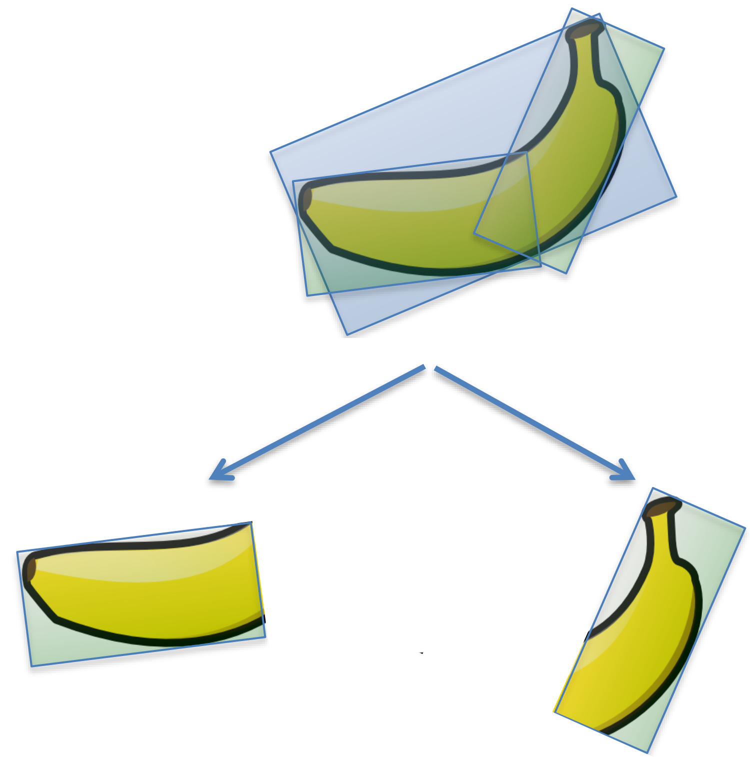
Bounding Volume Hierarchy

- "Look inside" the box:
  - Hierarchical structure
    - Root node completely encapsulates the object
    - Children give a "tighter fit" for the shape
    - Recursive / iterative algorithms to construct BVHs



Parent-Child Relationship

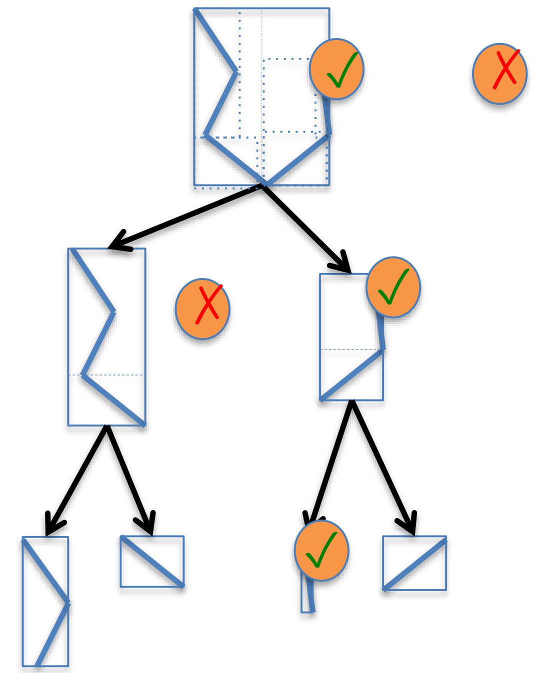
- Higher-level volumes may not contain their children volumes
  - higher level node contains the child's ***geometry***
  - best fit is the target
- Children volumes can intersect



(OBB hierarchy)

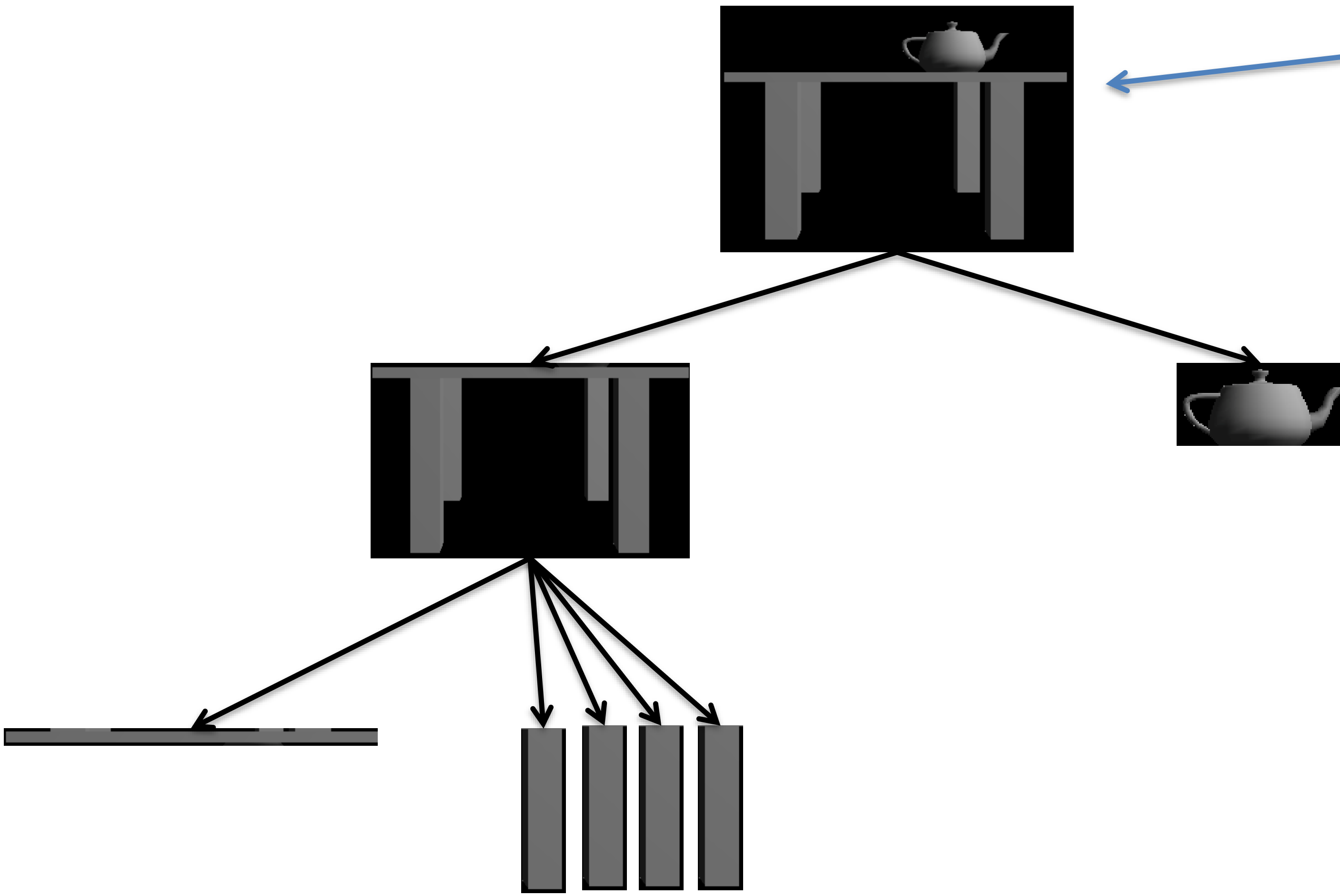
Bounding Volume Hierarchy-Based Collision Detection

- Given two BVH’s
  - If root volumes do not overlap
    - Return **False**
  - Else (may overlap)
    - Test recursively all pairs of children



(In this example the second shape is simply a sphere)

Bounding Volume Hierarchy and Scene Graph



Nodes
define
bounding
volume

Scene Graphs as Bounding Volume Hierarchies

Advantages:

- BHVs can be easily built from SGs

Disadvantages:

- Designers tend to group scene graph parts by function not by being close
  - A branch of light sources
- Can be too shallow / too deep

Collision Trees in jME

- jME automatically generates balanced bounding volume trees from geometries
  - Primarily for visualisation